**דו"ח פעולה**

**שלב 1**

**נושא**

בנית מודל חיזוי שיאפשר זיהוי חבילות תקינות וחבילות חריגות המעידות על התקפה בתעבורת הרשת.

**דאטה**

מקור: מאתר <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>

מאגר המידע מכיל:

* 397515 של מטא דאטה וסטטיסטיקות תעבורה של חבילות
* כ – 42 תכונות
* כ – 22 סוגי מתקפות.

עבור האימון אוחדו סוגי המתקפות לקטגוריה אחת, כך שהמודל יזהה חבילה תקינה normal או חבילה חריגה other, (תקינה = 1, חריגה = 0).

**שלב 2**

**סטנדרטיזציה ונרמול**

לאחר שלב זה מאגר האימון מכיל כ 120 תכונות. אימון המודל הינו ארוך מאוד לכן נצמצם:

**בחירת תכונות**

פונקציה 1: Missing Values

מציאת תכונות עם חלקים חסרים (Nan) מעל סף ספציפי (60%) והסרתם.

פונקציה 2: Collinear Features

Collinear features הן תכונות עם התאמה גבוהה אחת עם השנייה. בלמידת מכונה הדבר מוביל להורדה בכלליות הביצועים על מדגם הבדיקה בעקבות שונות גבוהה ופרשנות מופחתת של המודל.

השיטה מוצאת תכונות קולינאריות המבוססות על ערך מקדם המתאם שצוין. עבור כל זוג של תכונות מתואמות, הוא מזהה אחת מהתכונות להסרה (מכיוון שיש להסיר אחת מהן בלבד).

הדמיה מסודרת שאנחנו יכולים לעשות עם המתאמים הוא heat map. ההדמיה מציגה את הקורלציה של כל התכונות וכן את כל התכונות שיש להן לפחות קורלציה אחת מעל הסף (0.98).

פונקציה 3: Zero Importance Features

מוצאת תכונות בעלות חשיבות אפסית בהתאם למודל הלמידה של מדגם הדרגתי (GBM).

בעזרת מודלים מבוססי-עץ של לימוד מכונה, כגון boosting ensemble, אנו יכולים למצוא חשיבות ​​תכונות. הערך המוחלט של החשיבות אינו חשוב כמו הערכים היחסיים, בהם אנו יכולים להשתמש כדי לקבוע את התכונות הרלוונטיות ביותר למשימה. אנו יכולים גם להשתמש בחשיבות תכונות עבור בחירת תכונות על ידי הסרת תכונות שחשיבותן אפס. במודל מבוסס עץ, התכונות עם חשיבות אפס אינן משמשות לפיצול כל צומת, ולכן אנו יכולים להסיר אותן מבלי להשפיע על ביצועי המודל.

FeatureSelector מוצא את חשיבות התכונה באמצעות gradient boosting machine מספריית LightGBM. החשיבות ​​של התכונה הוא בממוצע מעל 10 ריצות אימון של ה- GBM כדי להקטין את השונות. כמו כן, המודל מאומן באמצעות עצירה מוקדמת עם סט בדיקה (יש אפשרות לכבות את זה) כדי למנוע overfitting על נתוני האימון.

gradient boosting machine זהו מודל אנסמבלי המשלב בתוכו מודלים פשוטים וכך מגיע לתוצאה מדויקת, במימוש שלנו הוא משלב עצי החלטה כאשר תכונה בחשיבות 0 אם באיטרציה האחרונה (בד"כ עשירית) היא לא מהווה צומת בעץ כך שלא משפיעה על המודל.

הערות: אימון gradient boosting machine הינו סטוכסטי, כלומר חשיבות תכונות תשתנה בכל הרצה של המודל. הדבר לא אמור להוות השפעה גדולה (התכונות החשובות ביותר לא יהפכו לפחות חשובות) אך זה ישנה את הסדר של חלק מהתכונות. הדבר עלול להשפיע בנוסף על זיהוי מספר התכונות בחשיבות אפס.

כדי לאמן את מודל הלמידה התכונות מקודדות בתחילה ב - one-hot. המשמעות היא שחלק מהתכונות שזוהו כחשיבות 0 עלולות להיות one-hot, והן עלולות להתווסף בתהליך אימון המודל.

פונקציה 4: Low Importance Features

מוצאת את התכונות הכי פחות חשובות שלא דרושות להשגת 99% מהחשיבות הכוללת.

הפונקציה משתמשת באחת השיטות של ניתוח רכיבים ראשיים Principal Components Analysis (PCA), שבה מקובל לשמור רק את ה – PC הדרושים לשמירה על אחוז מסוים השונות (כגון: 95%).

פונקציה 5: Single Unique Value Features

מציאת כל התכונות שיש להן ערך יחיד. תכונה בעלת ערך יחיד אינה שימושית עבור למידת חישובית מכיוון שיש לה אפס שונות.

למידע נוסף: <https://towardsdatascience.com/a-feature-selection-tool-for-machine-learning-in-python-b64dd23710f0>

פלט 5 הפונקציות על מאגר המידע:

Read 397515 rows.

0 features with greater than 0.60 missing values.

0 features with a single unique value.

11 features with a correlation magnitude greater than 0.9800.

53 features with zero importance after one-hot encoding.

48 features required for cumulative importance of 0.99 after one hot encoding.

67 features do not contribute to cumulative importance of 0.99.

76 total features out of 115 identified for removal after one-hot encoding.

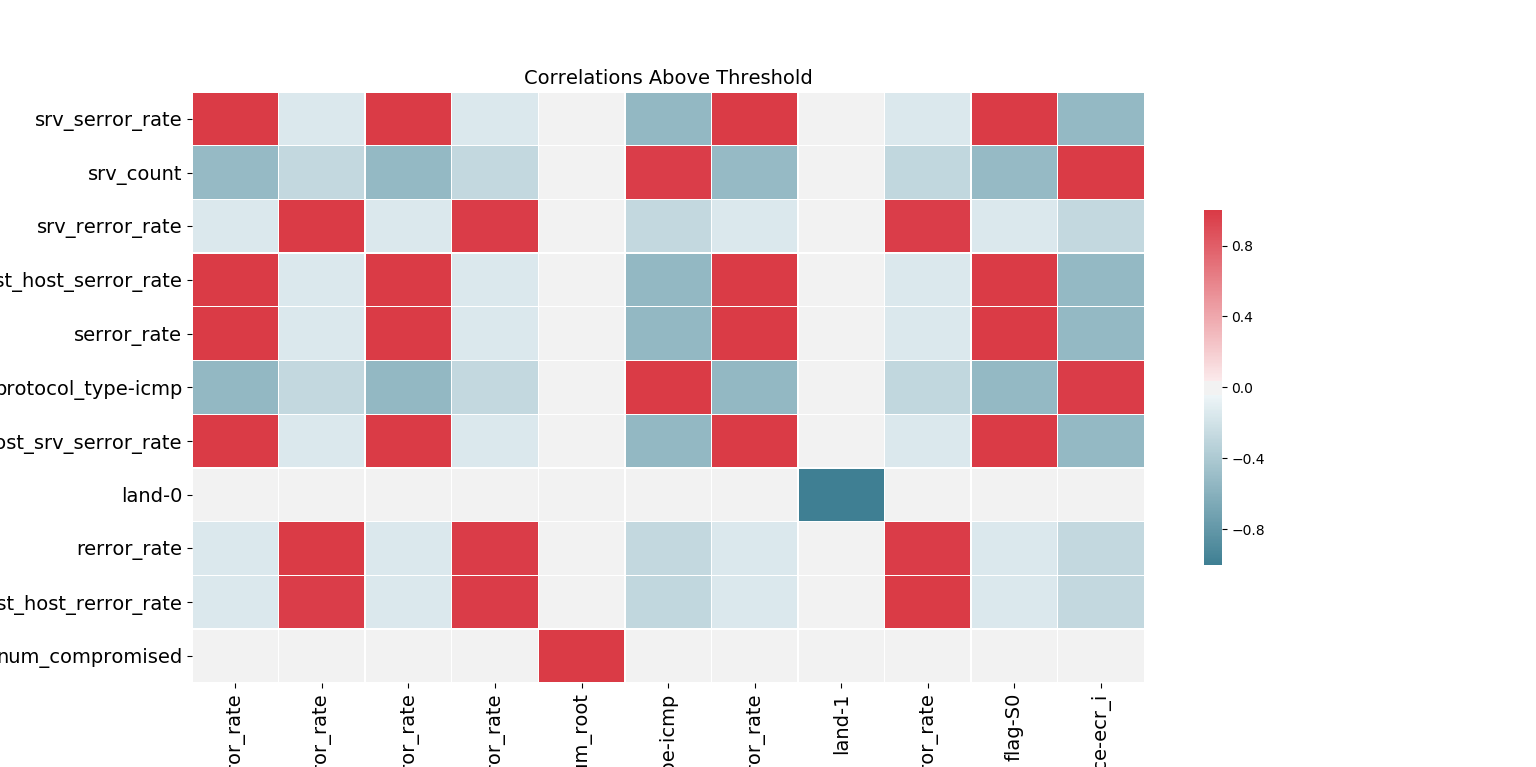
49 features required for 0.99 of cumulative importance

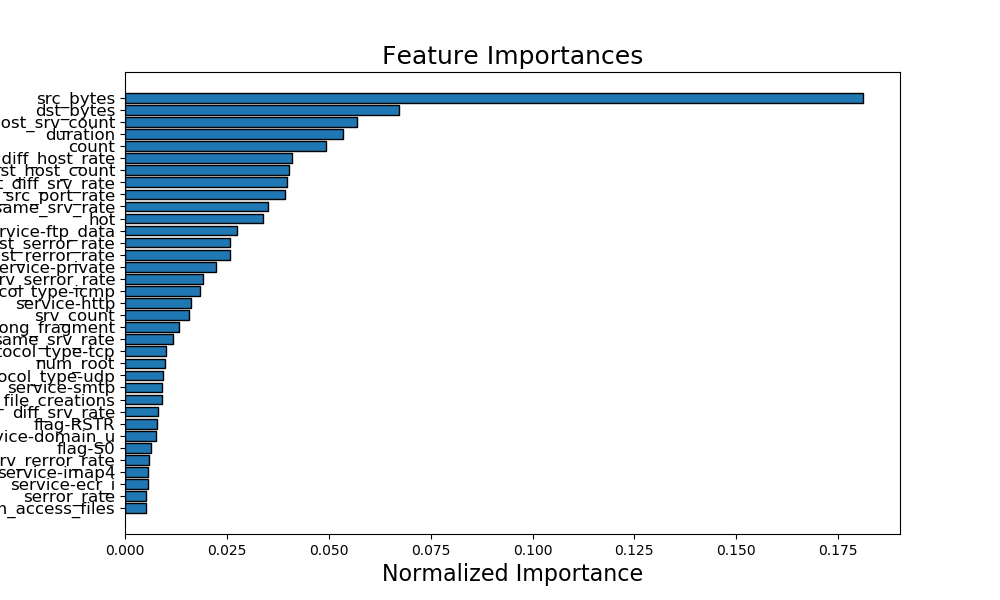
Removed 76 features.

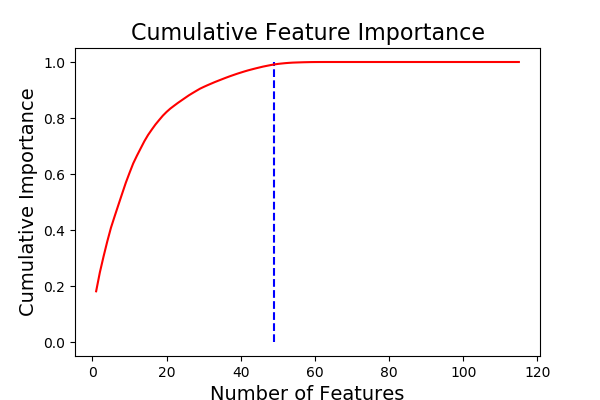
Original Number of Features 115

Final Number of Features: 39

דיאגרמה 1: קורלציה ליניארית בין כל התכונות (hit – map)

דיאגרמה 2:קורילציה ליניארית מעל סף 0.98

דיאגרמה 3: התכונות החשובות ביותר (מנורמל) בסדר יורד. 

דיאגרמה 4: חשיבות מצטברת לעומת מס' תכונות. הקו האנכי מצויר בסף חשיבות מצטברת. 

סט תכונות נבחר:

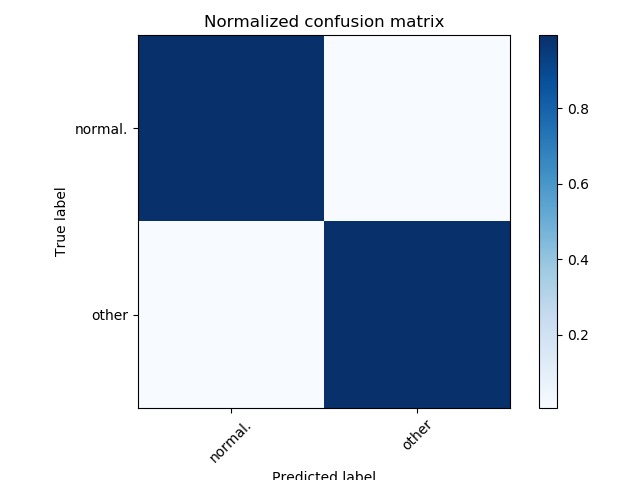
Features final set: ['duration', 'src\_bytes', 'dst\_bytes', 'wrong\_fragment', 'hot', 'num\_failed\_logins', 'num\_compromised', 'num\_file\_creations', 'num\_access\_files', 'count', 'srv\_count', 'serror\_rate', 'rerror\_rate', 'same\_srv\_rate', 'diff\_srv\_rate', 'srv\_diff\_host\_rate', 'dst\_host\_count', 'dst\_host\_srv\_count', 'dst\_host\_same\_srv\_rate', 'dst\_host\_diff\_srv\_rate', 'dst\_host\_same\_src\_port\_rate', 'dst\_host\_srv\_diff\_host\_rate', 'protocol\_type-tcp', 'protocol\_type-udp', 'service-IRC', 'service-auth', 'service-domain\_u', 'service-eco\_i', 'service-ftp\_data', 'service-http', 'service-imap4', 'service-login', 'service-other', 'service-private', 'service-smtp', 'service-telnet', 'service-time', 'flag-RSTR', 'flag-S1']

**אלגוריתמים**

* **Logistic regression–** אימון מודל עם רגולציה של L2 ובחירת גמא אופטימלי מתוך 10 אופציות(ללא מימוש של k fold מאחר ומדגם האימון הינו כחצי מיליון נתונים והחלוקה מאטה עוד יותר את אימון המודל).

דיאגרמות:

* + confusion metrics עבור אימון על פי גמא אופטימלי



Predicted label

* + גרף השגיאה על פי log10 של גמא:

The Accuracy Error with Regulation

For gamma = 1e-06 The Accuracy Error is: 0.0068257096138526485

For gamma = 0.0001 The Accuracy Error is: 0.00680055343591468

For gamma = 0.01 The Accuracy Error is: 0.006817324221206622

For gamma = 1.0 The Accuracy Error is: 0.007135969141755072

For gamma = 100.0 The Accuracy Error is: 0.00861179824745295

For gamma = 10000.0 The Accuracy Error is: 0.012385224938157746

For gamma = 1000000.0 The Accuracy Error is: 0.0191773929814264

For gamma = 100000000.0 The Accuracy Error is: 0.019504423294620765

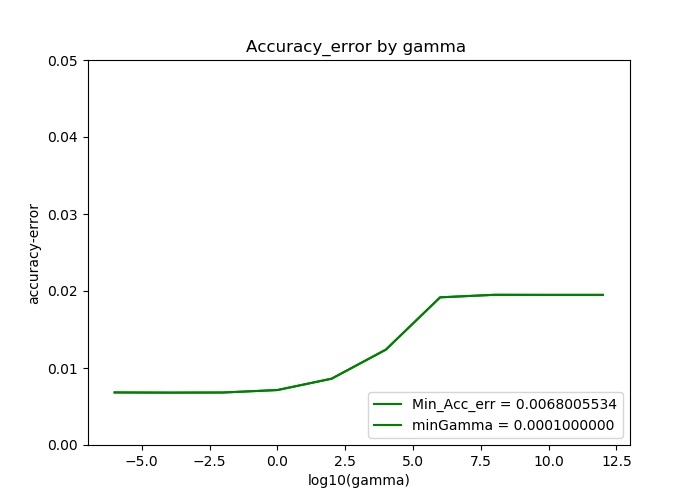
For gamma = 10000000000.0 The Accuracy Error is: 0.01949603790197474

For gamma = 1000000000000.0 The Accuracy Error is: 0.01949603790197474

Optimal gamma = 0.0001

ניתן לראות על פי הגרף את השגיאה על פי לוג 10 של גמא, ניתן לראות כי עבור

log10(gamma = 0.0001 = Optimal Gamma) השגיאה הינה הנמוכה ביותר (בפינה ימנית תחתונה של הגרף)



* + ROC Curve

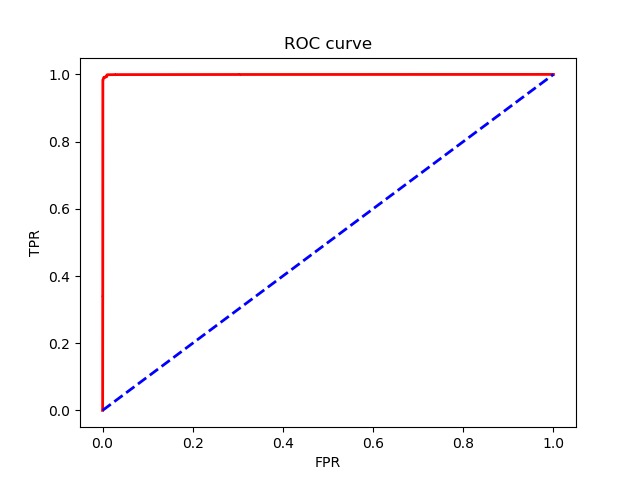
**Fpr** [0.00000000e+00 7.15717149e-05 7.15717149e-05 **...** 9.99928428e-01

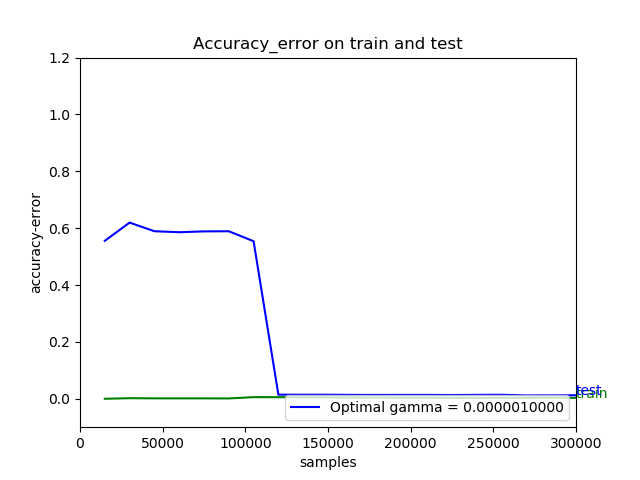
9.99964214e-01 1.00000000e+00]

**Tpr** [0. 0.00372932 0.00386493 **...** 1. 1. 1. ]

**Threshold** [2.00000000e+000 1.00000000e+000 1.00000000e+000 **...** 1.02208115e-250 5.09451353e-251 1.27681850e-251]

auc 0.9960315411231297

****



* + גרף השגיאה עבור מדגם

בדיקה קבוע ומדגם אימון

הולך וגדל – over-fitting

& under-fitting:

כחול – test, ירוק – train.

ניתן לראות לפי צורת הגרפים

כי המודל אינו over-fitting או

Under-fitting.

* **Support Vector Machine –** משמש לאימון המידע (training) בשלב האימון מתאימים מסווג שמפריד נכון ככל האפשר בין דוגמאות אימון חיוביות ושליליות המסווג יוצר מרווח גדול ככל האפשר בינו לבין הדוגמאות הקרובות לו ביותר בשתי הקטגוריות. כאשר מוצגת נקודה חדשה, האלגוריתם יזהה האם היא ממוקמת בתוך הגבול המגדיר את הקבוצה, או מחוצה לו. אנחנו משתמשות במסווג לא (בהכרח) לינארי ולכן נשתמש בגרעין(kernel).

בחירת הגרעין (kernel) מבין: Linear, Polynomial, Gaussian על ידי הערכת ביצועים.

דיאגרמות:

* + confusion metrics וכן classification report לכל אחד מהסוגים:

הסבר classification report:

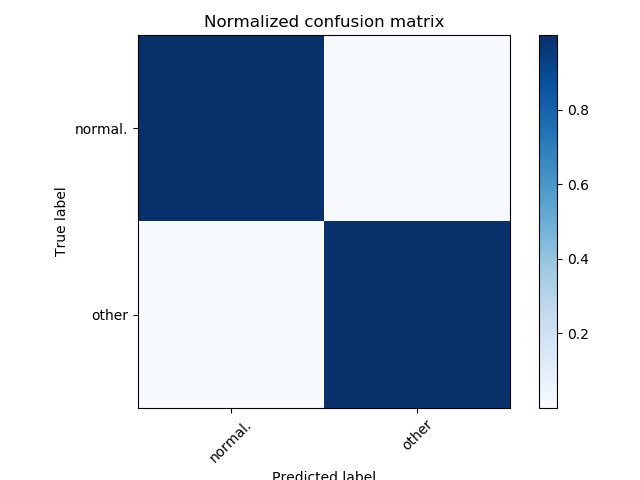
**- Precision** TP/TP+FP אחוז ההצלחה של המודל מתוך כל ההתקפות שחזה

**Recall**- TP/TP+FN מבין כל ההתקפות האחוז שהמודל זיהה

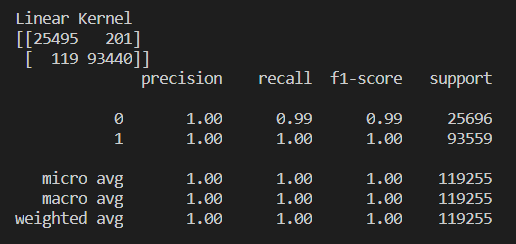
**f1-score -** מספר ההתקפות.

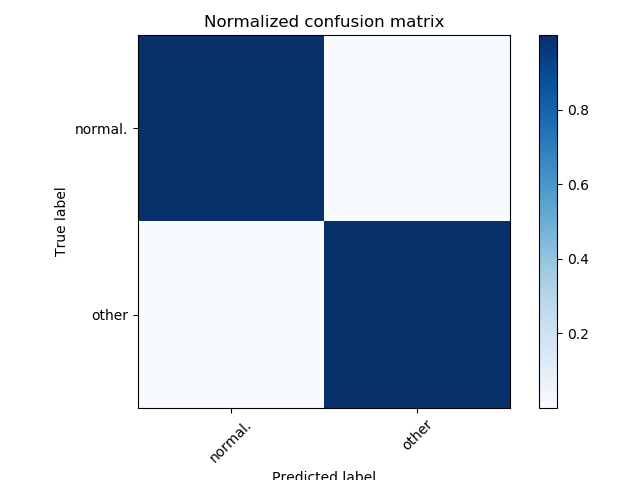
**Support –** מספר ההתקפות שהמודל חזה.

**Linear**



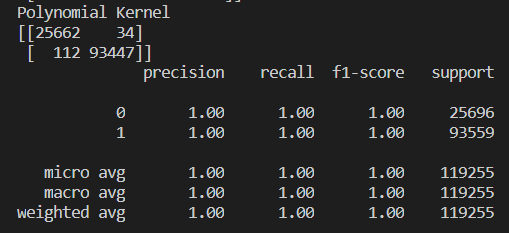
Predicted label



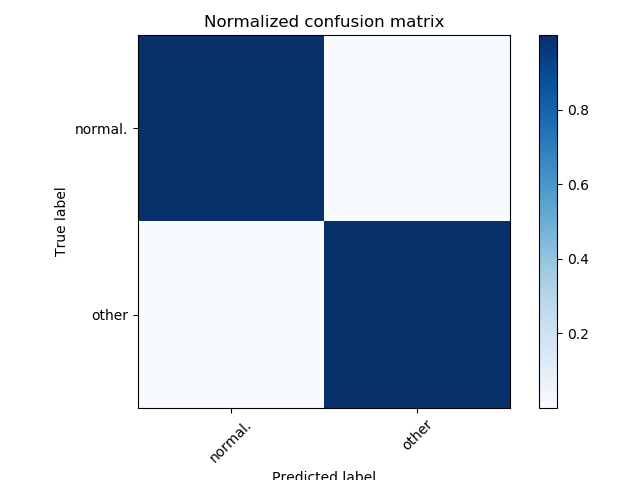


Predicted label

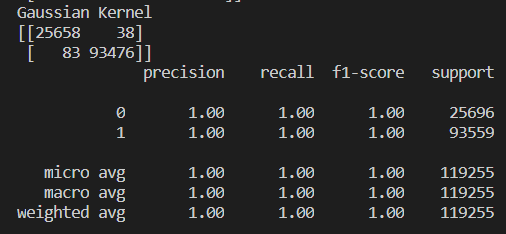
**Polynomial**



**Gaussian**

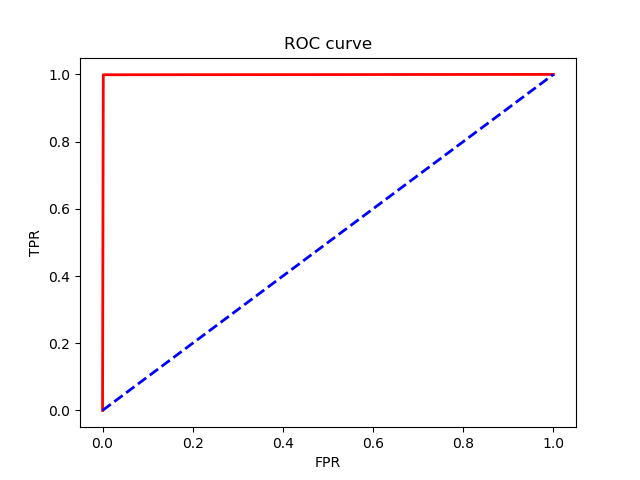


Predicted label

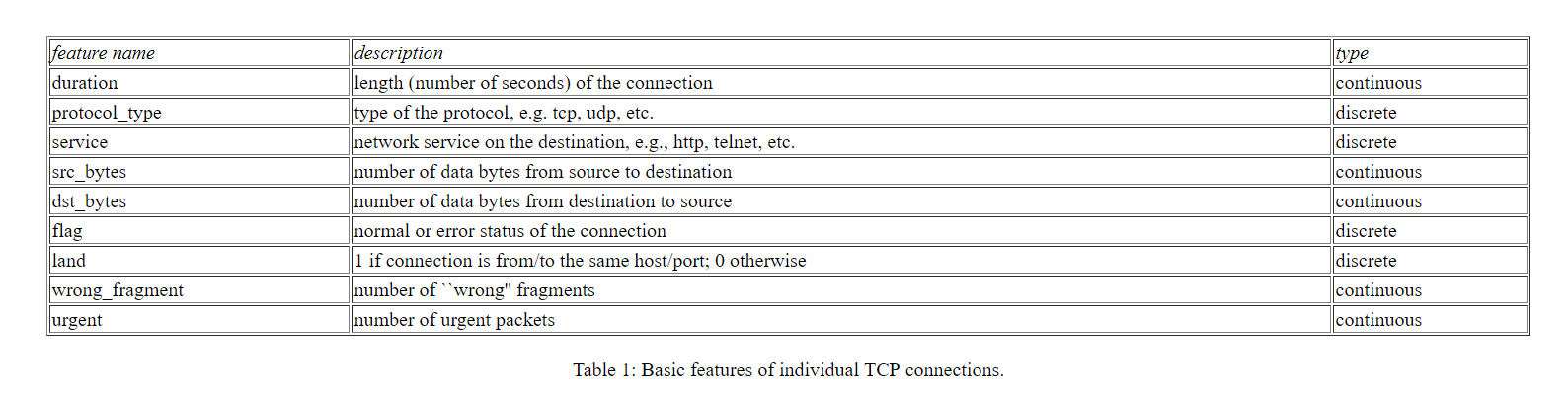


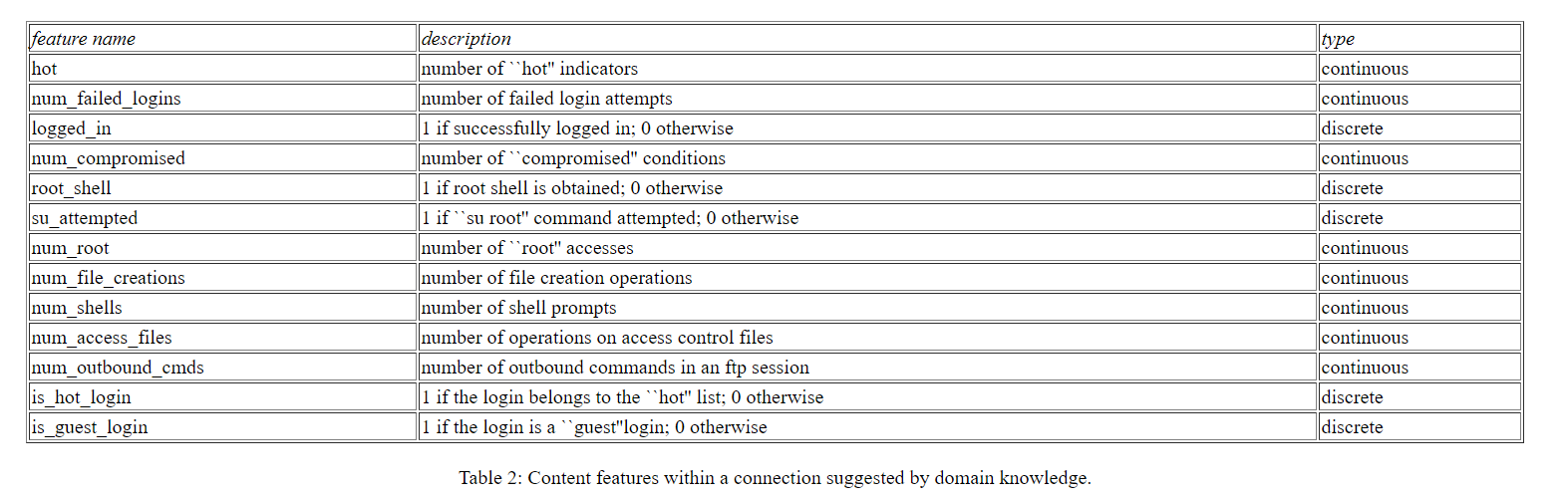
**בחרנו גרעין פולינומיאלי**

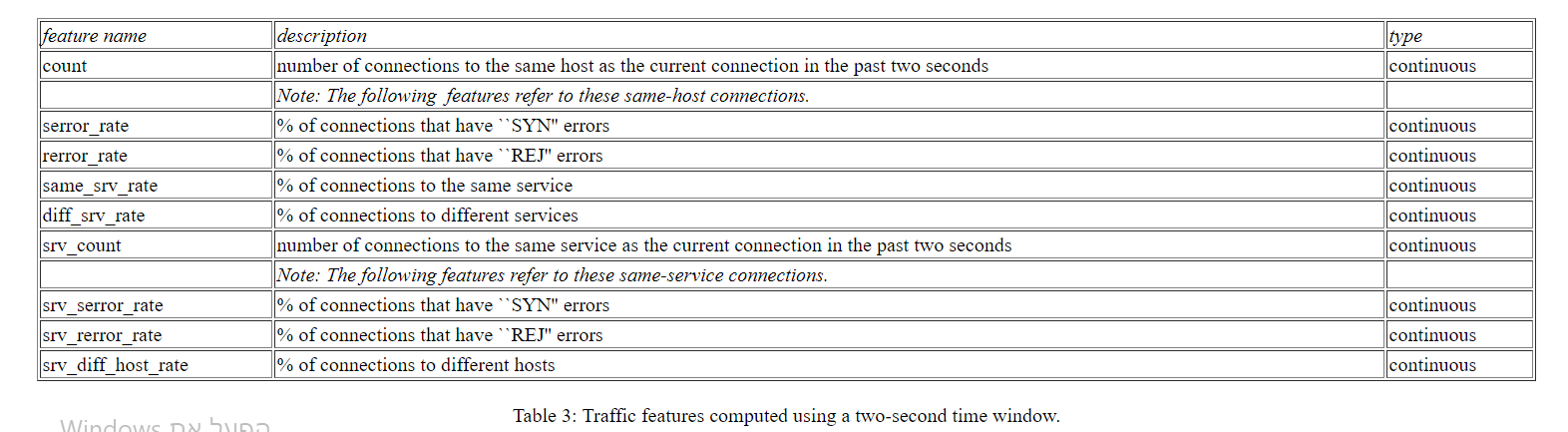
* ROC Curve עבור SVM עם גרעין פולינומיאלי:



נספח:

הסבר על תכונות הדאטה:





**A connection** is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol.  Each connection is labeled as either normal, or as an attack, with exactly one specific attack type.  Each connection record consists of about 100 bytes.